TileCal DCS Overview

Tentative Overview of the ATLAS configuration (1)

1. **Control and monitoring of Voltages:**
- **HV (power supply and PMT regulation)**
- **LV**
2. **Control and monitoring of calibrations systems:**
- **Cesium ?**
- **Laser**
- **Charge injection ?**

Tentative Overview of the ATLAS configuration (2)

3. **Control and monitoring of the cooling:**
- **Sharing of the whole system control ?**
4. **Control and Check-out of read-out electronic:**
- **Simple diagnostics to detect faulty componants (not yet approuved by Tilecal➔ next week)**
5. **First level safety station**

Control and Monitoring of Voltages (1)

HV for PMTs (10140) → two componants

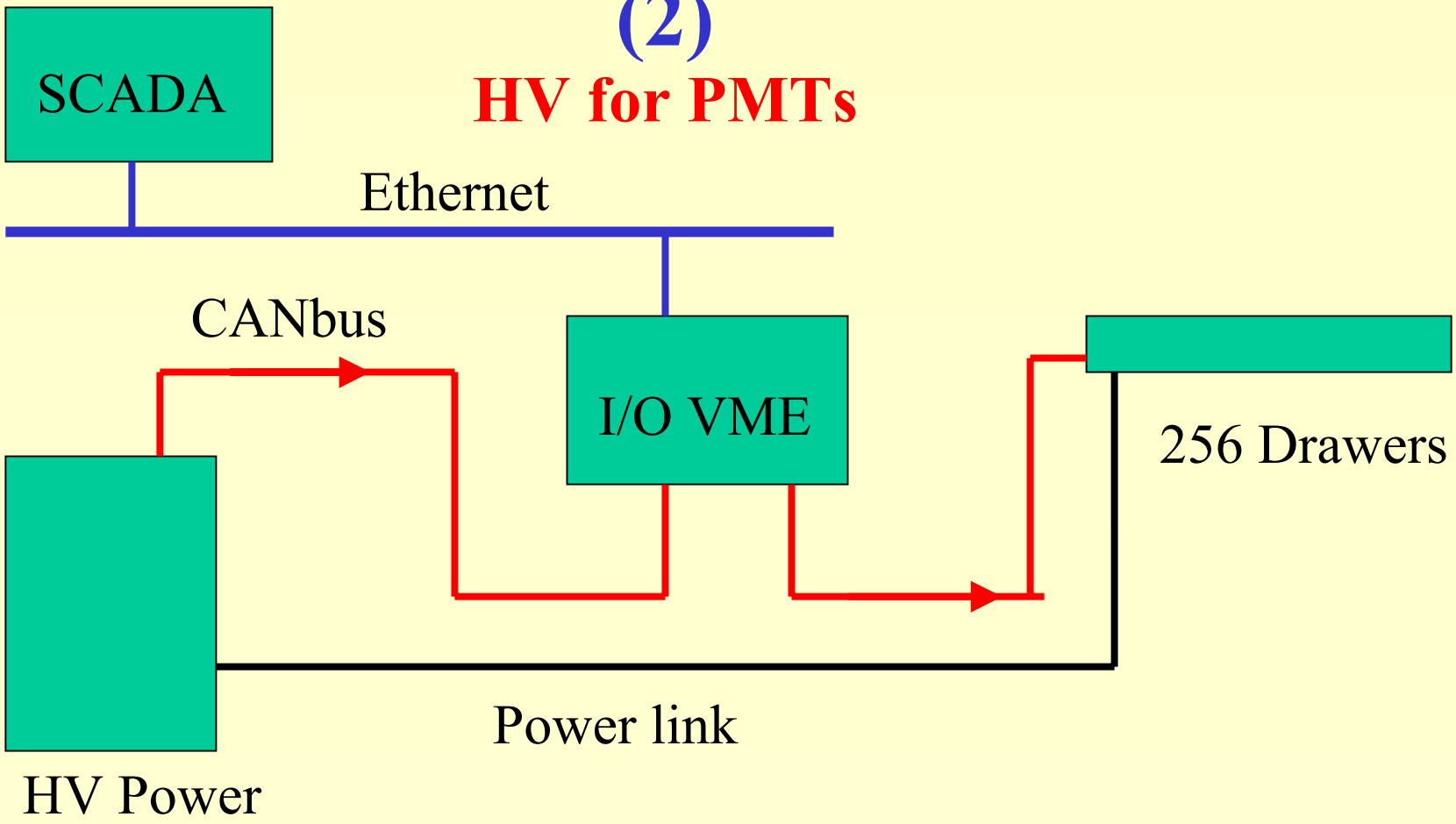
- **HV power supply (16 Crates) in USA15**
- **PMT regulation in « Drawers » through dedicated CANbus**

LV for

- **PMT HV regulation**
- **Electronic Read-out**

Control and Monitoring of Voltages (2)

HV for PMTs



Control and Monitoring of Voltages (3)

HV for PMTs

Need for communication between SCADA and VME

➔Tilecal Communication interface (TCI)

➔Control of HV Power Supplies through the same VME

➔ One full daisy chain (16 drawers) tested in 2002

Control and Monitoring of Voltages (4)

LV

Localisation of systems near the detector (finger)

➔Radiation tolerance avoid to use ELMB

➔Common solution with LAr but cost !

➔ Solution for ATLAS under study

(G. Blanchot and Y. Hruska)

Control and monitoring of calibration systems (1)

Laser

- Hardware under fabrication also use VME
- Level of integration in DCS to be checked

Cesium

System already fixed ➔ integration in DCS to be discussed

Charge injection

➔ difficult to integrate in DCS

Cooling Control

- First experience in testbeam (See Fernando talk)
- Full scale moke-up of hydraulic configuration of ½ barrel under construction with cooling unit
- Sharing of the monitoring and control not clear up to now → what part under subdetector responsability

Control and check-out of read electronic

- Tilecal read-out electronic in « drawers »
- Drawer fully checked and certified at assembly on a dedicated test-bench in Clermont using existing access channels

➔Possibility to use this experient to get a check-out system for the diagnostic of faulty componants

➔ to be discussed in Tilecal community

Activity planned in 2002

- **Full HV system checked**

(only 1 daisy chain = 16 « drawers »)

- **Start of Laser control and monitoring desig**

- **Control of the cooling moke-up**

- **LV after final hardware design**

Tilecal Communication Interface (1)

TCI developed to ensure communications between SCADA layer and VME based front/end system.

TCI does not contain any knowledge either of hardware or of SCADA

➔ designed as an general tool for creation of distributed multi-platform client-server application for hardware control.

Tilecal Communication Interface (2)

Command interface: **client issues a *synchronous* or *asynchronous* call requesting server to execute a command and to send back the result.**

- ***Data interface:*** **clients**
- **- subscribes on the hardware parameters (items) he want to receive,**
- **- specifying as well a condition which triggers data sending.**
- **The *callback* mechanism is used to deliver the data to the client.**

Tilecal Communication Interface (3)

- *Command interface:* client issues a *synchronous* or *asynchronous* call requesting server to execute a command and to send back the result.
- *Data interface*: client
- - subscribes on the hardware parameters (items) he want to receive,
- - specifying as well a condition which triggers data sending.
- The *callback* mechanism is used to deliver the data to the client.

Tilecal Communication Interface (4)

- **Client can request server to send data periodically (*polling*), when parameter was changed (*on data change*), on request (*refresh*).**
- ***Polling* and *on data change* mechanisms can be mixed.**

➔ **TCI based on CORBA 2.3 spec. (ORBacus 4)**

➔ **TCI server also in compact version of ORBacus/E for C++ (compact ORB for embedded applications).**

Tilecal Communication Interface (5)

TCI does not assume any knowledge of CORBA specifications by customer:

C++ classes wrap up the CORBA related code, providing user with a comprehensible interface.

The application developer has to dial with four abstract classes:

- **TCI_Init,**
- **TCI_Service,**
- **TCI_Command**
- **TCI_Client**

Tilecal Communication Interface (6)

and one concrete class

- TCI_CommandObj

User needs to derive concrete classes from abstract classes implementing application specific functions (server side)

- executing of commands,
- reading out of parameters,
- storage of delivered data on callback,
- application management functions (errors, shutdown…)

Tilecal Communication Interface (7)

The essential part of the whole system is Naming Service ➔ should be started before any other process.

- **at startup servers publish in Naming Service references for all commands and services they intend to offer for client applications.**
- **when the client application wants to be served by some service ➔ searches for a reference with desired name in the Naming Service**
- **then, using the information contained in this reference, establishes connection with server process.**

Tilecal Communication Interface (8)

TCI is implemented under Windows 95/98/NT/2000, Linux, LynxOS, BlueCat Linux.

It can be implemented under other UNIX platforms, supported by ORBacus as well (HP-UX, SUN Solaris, et al.).

One of advantages of TCI is quite high speed of data transfer → to be measured on the 2002 HV moke-up

Criticism of ATLAS DCS components

- **No fundamental criticism up to now**
- **Test-beam configuration quite stable**
- **Need more experience to really conclude**